9. Patrick

Program Name: Patrick.java **Input File: patrick.dat**

Patrick has been researching mathematical sequences for a special project and stumbled upon one called the "look-and-say" sequence. It is rather unique because it is not generated from an infinite application of a mathematical formula. Instead, the sequence is extended as if a human reader was describing the current number as a count of occurrences of each subsequence containing one specific digit. The starting number, like many numerical sequences is 1. If you were asked to describe the current number it would be one count of the digit one or "one one" which becomes the next number in the sequence 11. The new number contains "two ones" producing 21. Continuing the pattern, "one two, one one" produces 1211 then "one one, one two, two ones" produces 111221. This sequence is shown below along with several more numbers in the sequence.

1, 11, 21, 1211, 111221 as described above and sequence continues as follows:
312211 is "one three, one one, two twos, two ones"
13112221 is "one one, one three, two ones, three twos, one one"
1113213211 is "three ones, one three, one two, one one, one three one two, two ones"
31131211131221 etc.

How about starting with an arbitrary sequence of digits?

7440888 is "one seven, two fours, one zero, three eights"
17241038 is "one one, one seven, one two, one four, one one, one zero, one three, one eight"
1117121411101318 etc.

Can you help Patrick create a program to generated such sequences given an initial starting number?

Input: An unknown number of lines, greater than 1 and no more than 20. Each line contains 2 integers separated by a single space: N, the first number of a sequence with $0 \leq N \leq 999999999999999$ and P, the position of the desired number in the sequence with $1 \leq P \leq 25$. N will not contain a sequence of any single digit that is longer than 9.

Output: For each test case, display one line containing the P^{th} number of the sequence. Length of resulting number will not exceed 2000 digits.

Sample input:
```
1 9
7440888 3
2 13
5 1
```

Sample output:
```
31131211131221
1117121411101318
311311222113111231133211121312211231131112311211232221l2
5
```